GeoBench: a Geospatial Integration Tool for Building a Spatial Entity Matching Benchmark

Anthony Morana, Thomas Morel
UCBL
Lyon, France

Bilal Berjawi
LIRIS, UMR5205
F-69622, Lyon, France
bberjawi@liris.cnrs.fr

Fabien Duchateau
UCBL, LIRIS, UMR5205
F-69622, Lyon, France
fduchate@liris.cnrs.fr

ABSTRACT

In the last decade, a large market for location-based services and geospatial applications has emerged. Multiple cartographic providers propose their visualization tool for displaying points of interests. However, the data describing spatial entities is often incomplete and contradictory from one provider to another, thus limiting further applications such as geospatial data mining. Recent works in entity matching tackle this issue by discovering correspondences between spatial entities that refer to the same real world location. To evaluate and compare these works, we propose GeoBench, a tool which facilitates the building of a benchmark for spatial entity matching.

Categories and Subject Descriptors

Information Systems [**Data management systems**]: Information Integration; Information systems [**Information systems applications**]: Spatial-temporal systems

Keywords

Spatial Entity Matching, Spatial Integration, Spatial Data Quality, Entity Matching Benchmark

1. INTRODUCTION

Location-based services (LBS) are daily used in various applications, and cartographic providers play an essential role in displaying points of interest (POI) such as restaurants, hotels, and tourist places. A provider usually represents a POI using a spatial entity. The diversity and the multiple interconnections between cartographic providers is a source of noisy data [DPS98]. For instance, two providers may have incomplete and/or contradictory data for the same POI. This noise has a negative impact when users need to find reliable and relevant information. Besides, it constrains innovative research studies such as geospatial data mining and analysis. Recent entity matching approaches aim at tackling this issue by discovering correspondences between spatial entities which refer to the same POI. Some approaches may only consider the spatial information to detect correspondences [SKS$^+$10] while others exploit multiple criteria (e.g., name, type) to compute a similarity between two spatial entities [Olt07]. It is also possible to use training data for setting weights when averaging similarities computed by various measures [SGV06]. Such empirical approaches require a validation step, traditionally performed by experiments on real-world data.

To the best of our knowledge, there is no benchmark for evaluating spatial entity matching approaches. Indeed, the datasets used in the papers are not made fully available, for instance because of confidentiality issues [KSG07]. A few attempts are provided, such as this dataset about restaurants[1]. Yet, they cannot be exploited for various reasons. Some of them are not challenging (e.g., a simple equality metric applied to the phone numbers in the restaurant dataset discovers all the correct corresponding entities). Besides, a specific dataset may be required, for instance to include all POI types (e.g., restaurants, museums, mountains) or all entities from a given area. This lack of benchmark does not facilitate a fair and accurate comparison between the different spatial matching approaches. Besides, building an expertised dataset is costly in terms of human effort. We also argue that the properties of a dataset are useful, both for understanding why an entity matching approach is (not) effective, and for using appropriate training data when needed.

For those reasons, we believe that a benchmark for spatial entity matching is necessary. Thus, we present our tool GeoBench which assists users in building such benchmark by facilitating the discovery and the integration of corresponding spatial entities. The contributions in this paper are threefold: (i) algorithms for discovering correspondences between spatial entities, detecting the differences between corresponding entities, and merging them (ii) a tool named GeoBench which implements the aforementioned algorithms to help users building a benchmark, and (iii) a demonstration scenario for end-users to obtain a useful map showing their favourite places with complete and validated information resulting from the integration of different providers.

2. DESCRIPTION OF GEOBENCH

A spatial entity, which refers to a POI, is described with a set of attributes. We distinguish the **primary attributes** (identifier, name, type and coordinates) which are usually required for most providers and the **secondary attributes**

SIGSPATIAL'14, November 04 - 07 2014, Dallas/Fort Worth, TX, USA
ACM 978-1-4503-3131-9/14/11
http://dx.doi.org/10.1145/2666310.2666362.

[1]`http://www.cs.utexas.edu/users/ml/riddle/`

(e.g., address, phone number, website). A **correspondence** between two spatial entities means that these entities refer to the same POI. In our context, a **benchmark** is a set of pairs of entities, which have been expertized and tagged either as correct (i.e., a correspondence) or as incorrect. The rest of the section explains how GeoBench builds such benchmarks.

2.1 Overview of the tool

GeoBench's goal is to help users building a benchmark for evaluating spatial entity matching. We have designed GeoBench as a Web-based application so that it can run on various systems. It is accessible at the following URL:
`http://geobench.liris.cnrs.fr/`

Figure 1 illustrates the processes involved in GeoBench. The tool takes as input a source entity φ (selected by the user or at random) and a parameter κ (the maximal number of potentially corresponding entities proposed by the tool for each provider). Given these two inputs, GeoBench queries the cartographic providers using a blocking algorithm (see Section 2.2). For each provider, it obtains a set of κ potentially corresponding entities for the source entity φ. The matching algorithm is in charge of ranking the potentially corresponding entities to propose the correct one at the top (see Section 2.3). Next, we apply the algorithm for detecting the differences between the attributes of potentially corresponding entities (see Section 2.4). The potentially corresponding entities and the detected differences are presented to the user for validation. When the validation is done, the integration algorithm enables a semi-automatic merging of the corresponding entities, i.e., those which have been validated as correct (see Section 2.5). The user has the possibility to modify the attribute values suggested during the merging.

The whole process can be repeated by choosing new parameters (a source entity φ and a value κ). Thus, many merged entities can be generated and finally displayed on a customized map. At the end, this dataset is available as a SQL script and can be used as a benchmark with well-known evaluation metrics such as precision and recall [BBR11]. In the next parts, we focus on describing the four main algorithms at the core of GeoBench (blocking, matching, detecting differences and integrating).

2.2 Blocking Algorithm

Each provider owns millions of spatial entities. A blocking algorithm aims at quickly identifying a subset of entities among all those available. In our context, the blocking algorithm needs to select a few entities which likely represent the parameter φ. Intuitively, we could perform the blocking based on the coordinates of the source entity φ and include all entities within a radius. Yet, this is not sufficient for two reasons. First, even a small area (e.g., city centres) may contain thousands of entities, thus limiting the benefit of the blocking. Besides, largest POIs such as mountain ranges or parks may have their coordinates either in the center of the POI or at one of the entrance. GeoBench performs the blocking based on the spatial aspect, which is later refined according to the POI type or the POI name. More precisely, we define the blocking area by using the POI coordinates and a radius. The radius value depends on the POI type. For restaurants or hotels, the radius value is set to *"50 meters"* while it is equal to *"500 meters"* for a park. Note that this value could be adjusted or learned using the benchmark. Within this blocking area, we first need to obtain all entities which have the same type than φ. Each provider has its own hierarchy of labels for representing the existing types of POI. For instance, the *"lodging"* type for Google Maps corresponds to the *"accommodation"* type for Here. Except for Bing and TomTom which both have hierarchies containing hundreds of types, an alignment has been manually produced between the hierarchies of the other providers. This alignment enables us to run a first blocking query based on the type and area. Similarly, the second query obtains all entities of the blocking area whose name shares a token with φ's name. The result of the blocking is the union set of the two previous queries. This limited set of potentially corresponding entities can now be compared with the source entity φ using state-of-the-art matching techniques.

2.3 Matching Algorithm

The blocking algorithm has constrained the number of entities to be matched, and the matching process aims at computing a confidence score between each of those selected entities and the initial source entity φ. Note that we focus on the data level (entity matching), i.e., the matching at the schema/ontology level is out of scope. Although the study of schema matching and ontology alignment has generated many tools and approaches, the schemas of cartographic providers are sufficiently small and static to be manually matched. Currently, we also exclude POIs represented with a polygon. However, most providers still represent large POIs with a point, thus our algorithm is able to match them.

The challenge of the matching algorithm is to produce relevant confidence scores for ranking entities resulting from the blocking process. A confidence score close to 0 means that an entity is totally dissimilar to the source entity φ. A confidence score equal to 1 indicates that both entities are equivalent, according to the matching algorithm. Contrary to the blocking algorithm, which quickly identifies potential corresponding entities using three attributes, the matching algorithm is based on sophisticated but costly similarity measures applied to all attributes.

To compute the confidence score, we compute similarity values between the attributes of a blocked entity and those of the source entity φ. Let us describe the different attributes and how we compare them. The **coordinates** of two entities are compared according to the Euclidean distance. The least distance between both entities, the closer to 0 the similarity value for coordinates is. The Levenhstein measure is applied between the **names (or titles)** of each entity. This similarity measure computes the number of operations (e.g., adding a character) to transform a first string into the second string, and it is normalized into the range $[0, 1]$. It has been demonstrated that the Levenhstein measure is the most effective with regards to other string similarity measures [SGV06]. Using other types of similarity metrics could improve the results of the matching between two names, but the metrics based on a dictionary/ontology (e.g., Resnik) or on the analysis of bag of words (e.g., Jaccard) are costly in terms of computation. Besides, using several metrics to match the same attribute involves a new problem for combining smartly the different similarity values. The attributes corresponding to the concept **phone number** and **website** are also matched using the Levenhstein measure. The **address** attribute requires a pre-processing step to normalize the different formats. Comparing each of the individual ele-

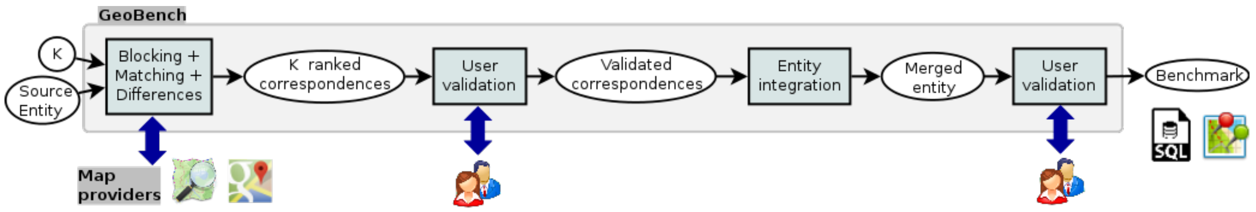

Figure 1: Overview of the processes involved in GeoBench

ments of an address (e.g., postcode, city, street name) would have the same drawback as using several similarity measures, i.e., it would be necessary to combine the individual similarity values into a global score for the address attribute. Thus, we decide to merge the individual elements of an address into one normalized element[2], so that the Levenhstein measure can be applied. The main advantage of such normalization is that a difference in the postcode value or in the street number value (which are common mistakes) does not strongly affect the similarity computed between two addresses. The last attribute, namely **type**, is crucial because it enables the limitation of the search space during the blocking process (see Section 2.2). The manual alignment of the type's hierarchies facilitates the computation of a similarity value between two types. Our similarity measure is inspired by Resnik's similarity, which consists in detecting the common matching ancestors of two concepts [Res99]. Other attributes are mainly specific to a given domain (e.g., *"type of meal"*, *"opening hours"*). Since we want GeoBench to be generic, such attributes are not used.

When all the individual scores have been computed, we need to compute the confidence score. A weighted average is traditionally used for combining the individual similarity values. GeoBench also combines them with this technique and it provides more weight to the most important attributes. Indeed, the secondary attributes such as phone or address may be missing for a provider. Thus, we tune their weight to one-third, while the primary attributes have a weight equal to two-thirds. A decision step is finally required to select the correspondences, i.e., the entity which corresponds to φ. Various methods such as a threshold or the top-K enables this automatic selection [BBR11]. In our context, proposing the top-K correspondences to the user is the most relevant choice because the user has to manually verify these suggested correspondences. Besides, it is easier for an end-user to tune a parameter related to a number of propositions to be validated (κ in our case) rather than tuning a mysterious threshold value. At the end of the matching process, GeoBench outputs for each provider an ordered list of at most κ entities which are ranked according to their confidence score, and the user validates those entities which may correspond to the source entity φ.

2.4 Detecting differences

This step aims at classifying the terminological and spatial differences between the attributes of two entities. For instance, the attribute *"city"* has a value equal to *"Dallas"* for the source entity φ, and the same attribute of a corresponding entity has a value *"Dallas, TX"*. Although both character strings are not identical, the concept represented by both values is the same (the city of Dallas, Texas). In that case, there should not be a difference between the two values. The motivation for identifying these differences is threefold. First, remind that the datasets built with GeoBench are used for testing spatial entity matching algorithms. Thus, it is important to be able to characterize a dataset: one may contain only entities from a specific type while another includes many corresponding entities with totally different names. Secondly, the user validates all suggested entities, including the incorrect ones. When a spatial entity matching algorithm makes errors (false positives and false negatives), it is useful to have clues about the reason(s) which caused these errors. The detection of differences is a first step towards the understanding of errors by indicating which attribute(s) of the correspondence led to an error. Last, many spatial matching algorithms rely on machine learning techniques [Olt07, SGV06]. The selection of training data is a complex issue, but information about possible differences is helpful in this case.

The detection of differences is mainly based on the similarity values computed between two attributes. For instance, the coordinates are considered different when the Euclidean distance reaches a given threshold (depending on the POI type). Two terminological attributes (e.g., names, phones) are suggested to be different when their similarity value computed with Levenshtein measure is below a threshold. This detection process of the differences can be improved using statistics for instance, but we rely on the expertise and preferences of the user, who validates all suggested differences.

2.5 Integrating Corresponding Entities

After user validation, GeoBench offers the possibility to merge corresponding entities into a new integrated entity. This integration is a common task in applications such as crisis management, datawarehousing or mashup creation. The integration process relies on the discovery of correspondences, and it aims at generating an integrated entity. The main issue is to select the values for each attribute of the integrated entity. Usually, there is not a single solution, and it may depend on user preferences. Yet, it is interesting for a benchmark to evaluate if merging algorithms make the same choice as an expert. Thus, GeoBench includes a process for determining which value is the most suitable for each attribute of the integrated entity. Our intuition is to propose the value which is the less dissimilar to others. To compute this dissimilarity, we use the Levenshtein score, i.e., the number of operations needed to transform a string into another. The idea consists of adding all Levenhstein scores of a given value, since each of these scores reflects a degree of dissimilarity with another value. Consider three corresponding entities from Geonames, Here and Google Maps with respective titles *"Eiffel Tower"*, *"58 Tour Eiffel"* and *"Tour*

[2]The normalization of the address is as follows: street number, street name, postcode, city, country.

Eiffel". The Levenshtein score between *"Eiffel Tower"* and *"58 Tour Eiffel"* equals 12, the score between *"Eiffel Tower"* and *"Tour Eiffel"* is 11 and the transformation of *"58 Tour Eiffel"* into *"Tour Eiffel"* requires 4 operations. The dissimilarity score of the value *"Eiffel Tower"* is therefore equal to 23 (12 + 11), the dissimilarity score of *"58 Tour Eiffel"* equals 16 and the one for *"Tour Eiffel"* is 15. In this example, GeoBench suggests to the user the integrated value with the minimal dissimilarity score (*"Tour Eiffel"*). A specific case is for the coordinates attributes, which has no meaning as a character string. Thus, we use the same idea applied to the distance in meters. The suggested value is the one which is the closest to all other values. Users are free to modify the value suggested by our algorithm. When the merging is finished, all integrated entities are shown on a map. Users can download data about the corresponding and the integrated entities in SQL format so that they can be used for benchmarking entity matching approaches.

3. DEMONSTRATION SCENARIO

This section describes the use case which will be demonstrated at SIGSPATIAL. GeoBench is useful for researchers in data integration for constructing spatial entity matching benchmarks, but we have chosen to present a scenario interesting for end-users, namely the generation of a customized and complete map. Currently, we consider one provider as the source (Geonames, the largest database of place names), on which users select a source entity. Next, GeoBench discovers, for the source entity, the corresponding entities on the target providers. We currently offer two target providers (Google Maps and Here). In addition to online testing of GeoBench, we also provide a video tutorial and a few screenshots of this demonstration scenario[3].

Zoey plans to travel to Dallas, Texas. To obtain complete and accurate information about the hotels in Dallas, she uses GeoBench. She queries for *"Dallas"* with POI type equal to *"lodging"*. The maximum number of results by provider is set to *5* because Zoey is confident in the quality of the tool. The first suggestion on Geonames is *"Hyatt Regency Hotel"*, which is conveniently located for Zoey. However, the source provider does not specify the phone number and the address for this hotel. GeoBench runs the matching process using the algorithm detailed in Section 2.3 and it first suggests potentially corresponding entities for the provider Here. Zoey quickly confirms that the single proposed entity (with a confidence score at 67%) corresponds to the POI *"Hyatt Regency Hotel"*. Note that the differences between the Geonames entity and the Here entity are also computed, but they are not useful in this scenario. Zoey validates this entity as the corresponding one for the Here provider. Luckily, she now knows the full address because it is provided by Here. But Zoey still does not have the phone number of the hotel. She runs the matching for the Google Maps provider and GeoBench displays one entity with a confidence score at 48%. Although this score is not high, the proposed entity is the one for *"Hyatt Regency Hotel"* and it includes the phone number and the website. She can now start the merging process to integrate all information from the three providers into a single complete entity. GeoBench lets Zoey decide which values should be included in the integrated entity, but it facilitates the choice by preselecting values based on the algorithm described in Section 2.5. Once the merging step is done, GeoBench displays the integrated entity on a map (currently using Google Maps). Zoey can repeat the process for adding more locations (e.g., other hotels, restaurants) to obtain a customized map with her favourite places. Note that for benchmark construction, the results of the integration is available as a SQL script too.

[3]`http://geobench.liris.cnrs.fr/help.html`

4. CONCLUSION

In this paper, we have proposed GeoBench, which facilitates the building of benchmarks for spatial entity matching. It is based on a simple but effective matching algorithm, and it enables the detection of differences between the values of attributes from different entities as well as the merging of these values. During the demo session, users will be able to search for their favourite locations, check and validate suggested corresponding entities from other providers, merge their information and visualize the result of this integration process on a new customized map. The scenario described in the paper strongly advocates for tools such as GeoBench. On the three providers, none of them has complete information about the *"Hyatt Regency Hotel"*: Geonames only includes the name, Here provides the correct address of the hotel and Google Maps shows the phone number and the website, but an incomplete address. Thus, our tool is a benefit for end-users, but also for researchers in GIS, data cleaning, data integration and machine learning. As a perspective, we intend to build a benchmark composed of datasets with specific properties so that spatial entity matching algorithms can be compared using the same baseline.

5. REFERENCES

[BBR11] Zohra Bellahsene, Angela Bonifati, and Erhard Rahm. *Schema Matching and Mapping.* Springer-Verlag, Heidelberg, 2011.

[DPS98] Thomas Devogele, Christine Parent, and Stefano Spaccapietra. On spatial database integration. *International Journal of Geographical Information Science*, 12(4):335–352, 1998.

[KSG07] Hyunmo Kang, Vivek Sehgal, and Lise Getoor. Geoddupe: A novel interface for interactive entity resolution in geospatial data. In *IV*, pages 489–496, 2007.

[Olt07] AM Olteanu. A multi-criteria fusion approach for geographical data matching. *International Symposion in Spatial Data Quality*, 2007.

[Res99] Philip Resnik. Semantic similarity in a taxonomy: An information-based measure and its application to problems of ambiguity in natural language. *Journal of Artificial Intelligence Research*, 11:95–130, 1999.

[SGV06] Vivek Sehgal, Lise Getoor, and Peter Viechnicki. Entity resolution in geospatial data integration. In Rolf A. de By and Silvia Nittel, editors, *GIS*, pages 83–90. ACM, 2006.

[SKS+10] Eliyahu Safra, Yaron Kanza, Yehoshua Sagiv, Catriel Beeri, and Yerach Doytsher. Location-based algorithms for finding sets of corresponding objects over several geo-spatial data sets. *International Journal of Geographical Information Science*, 24(1):69–106, 2010.